(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 195 20 987 A 1**

(51) Int. Cl.[6]: **H 03 M 13/12**

(21) Aktenzeichen: 195 20 987.7
(22) Anmeldetag: 8. 6. 95
(43) Offenlegungstag: 12. 12. 96

DE 195 20 987 A 1

(71) Anmelder:
Siemens AG, 80333 München, DE

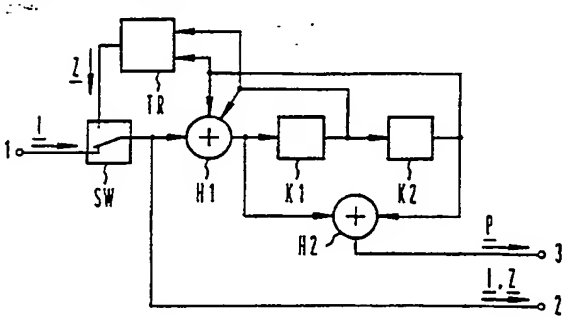
(72) Erfinder:
Hagenauer, Joachim, Prof., 82229 Seefeld, DE;
Burkert, Frank, 80796 München, DE

(56) Entgegenhaltungen:
O. JOERSSEN und H. MEYR »Terminating the trellis of turbo-codes« in: Electronics Letters, 4. Aug. 1994, S.1285-1286;

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Verfahren zur Terminierung des Trellis bei rekursiven systematischen Faltungscodes

(57) Der Coder (COD) enthält einen Terminator (TR), der nach der Codierung einer Informationssequenz ($\underline{I}$) aufgrund des Coder-Gedächtnisses (K1, K2) eine Terminierungssequenz ($\underline{Z}$) erzeugt, die das Coder-Gedächtnis (K1, K2) in einen gewünschten Sollzustand stellt.



DE 195 20 987 A 1

Beschreibung

Die Erfindung betrifft ein Verfahren zur Terminierung des Trellis bei rekursiven systematischen Faltungscodes und einen geeigneten Coder zur Durchführung des Verfahrens.

Zur Datensicherung werden unterschiedliche Codierungsverfahren verwendet. Bei diesen Verfahren werden von den zu übertragenden Informationsbits Kontrollbits abgeleitet, die ebenfalls übertragen werden. Mit Hilfe dieser Kontrollbits ist es empfangsseitig möglich, gefälschte Informationsbits zu erkennen und zu korrigieren.

Bei Faltungscodes wurden sogenannte Tail-biting-Verfahren entwickelt, um für die letzten zu codierenden Informationsbits den ohne Zusatzmaßnahmen geringer werdenden Fehlerschutz zu erhöhen. Dieses Verfahren ist in IEEE Trans. on Comm., vol. COM-34, no.2, 1986 von H.H. Ma, J.K. Wolf, "On Tail Biting convolutional codes" angegeben.

Die bekannten Tail-biting-Verfahren für nicht rekursive Faltungscodes lassen sich jedoch nicht auf rekursive systematische Faltungscodes übertragen. Diese Codegruppe ist jedoch aufgrund ihrer Eigenschaften besonders als Komponentencodes für Mehrkomponentencodes, die sogenannten "Turbo-Codes", von außerordentlicher Bedeutung. Diese Codegruppe ist in den folgenden Literaturstellen ausführlich beschrieben:

— C. Berrou, "Near Shannon limit error-correcting and decoding: Turbo-Codes (1)", Proc. ICC'93, May 1993;

— Demande de brêvet européen, N. de publication: 0 511 141 Al, Inventeur: C.Berrou, "Procédé de codage correcteur d'erreurs á moins deux codages convolutifs systématiques en parallèle, procédé de décodage itératif, module de décodage et décodeur correspondants"

— J. Hagenauer et al, "Iterative ("Turbo") decoding of systematic convolutional codes with MAP and SOVA algorithms", ITG Fachtagung "Codierung", München, Okt. 1994

— J. Hagenauer, L. Papke, "Decoding "Turbo"-Codes with the Soft Output Viterbi Algorithm (SOVA)", 1994 International Symposium on information theory, Trondheim, 1994.

Aus IEEE, Globecom 1994, Seite 1298 bis 1303, Robertson:
"Illuminating the Structure of code and decoder of parallel concatenated recursive systematic (Turbo) codes" ist die "Trellis Termination" auch für rekursive Codes beschrieben, jedoch ohne eine Realisation anzugeben.

Beim "Zero Tail-Verfahren" wird einem Coder für einen nicht rekursiven Faltungscode nach der Codierung der Informationsbits eine Sequenz von Zusatzbits zugeführt, die ihn in den gewünschten Sollzustand zwingt, d. h. das Trellis wird terminiert. Dieser Umstand wird vom Decoder mitbewertet, so daß der Fehlerschutz für die letzten Informationsbits eines Datenblokkes erhöht wird.

Aufgabe der Erfindung ist es, ein einfach zu realisierendes Verfahren zur Terminierung des Trellis bei rekursiven systematischen Faltungscodes anzugeben. Außerdem ist ein geeigneter Coder anzugeben.

Diese Aufgabe wird durch das in Anspruch 1 angegebene Verfahren gelöst.

In einem unabhängigen Anspruch wird ein geeigneter Coder beschrieben.

Vorteilhafte Weiterbildungen des Verfahrens und des Coders sind in den abhängigen Ansprüchen angegeben.

Besonders einfach wird das Verfahren, wenn im Sollzustand alle Speicherstufen des Coders dieselbe Information, die log. Null oder die log. Eins aufweisen. Dieser Zustand kann mit der Anfangszustand zu Beginn der Codierung eines Datenblocks übereinstimmen.

Die Terminierung wird besonders einfach, wenn aus den Zuständen des Coder-Gedächtnisses, z. B. von Speicherstufen, jeweils das nächste dem Coder zuzuführende Zusatzbit der Terminierungssequenz errechnet wird.

Bei der Realisation eines Turbocodes kann die Terminierung des Trellis bei einem Komponentencoder bei mehreren, aber auch bei allen Komponentencodern durchgeführt werden.

Das erfindungsgemäße Verfahren soll anhand von Ausführungsbeispielen näher beschrieben werden.

Es zeigen:

Fig. 1 einen Coder für einen rekursiven systematischen Code mit Terminator,

Fig. 2 zeigt ein Prinzipschaltbild zur Realisierung von Mehrkomponentencodes (Turbo-Codes) mit Terminator,

Fig. 3 zeigt eine erste Variante zur Realisierung von Mehrkomponentencodes (Turbo-Codes) mit Terminator und

Fig. 4 zeigt eine weitere Variante zur Realisierung von Mehrkomponentencodes (Turbo-Codes) mit Terminator.

In **Fig.** 1 ist ein Coder für einen rekursiven systematischen Code mit zwei binären Speicherstufen K1, K2 sowie zwei Modulo-2-Addierern H1 und H2 dargestellt. Über einen Dateneingang 1 und einen Umschalter SW gelangt jeweils eine Informationssequenz $\underline{I} = I_1, I_2 \ldots I_L$ bitweise zum Informationsausgang 2 und gleichzeitig zu einem Eingang des ersten Modulo-2-Addierers H1, dem außerdem die in den Speicherstufen K1, K2, dem Coder-Gedächtnis, vorliegenden Bits zugeführt werden. Das Ergebnis der Modulo-2-Addition wird dem Dateneingang der ersten Speicherstufe K1 zugeführt. Durch eine weitere Modulo-2-Addition der Modulo-2-Summe am Ausgang des ersten Modulo-2-Addierers H1 und der am Ausgang der zweiten Speicherstufe K2 anliegenden Information werden Kontrollbits P generiert und am Kontrollausgang 3 abgegeben. Die an den Ausgängen 2 und 3 anliegenden Codesymbole (Bits) werden in der Regel bitweise verschachtelt ausgesendet. Der Coder arbeitet in bekannter Weise mit einem Bittaktsignal, das in diesem Prinzipschaltbild nicht dargestellt ist.

Nachdem die Information eines Datenblockes codiert ist, besteht das Problem nun darin, das Coder-Gedächtnis, die M in den Speicherstufen K1 und K2 gespeicherten Daten, insgesamt sind bei binären Speichern 2^M verschiedene Variationen möglich, so zu verändern, daß eine bestimmte Sollzustand, beispielsweise die Anfangszustand "log. Null" für alle Speicherstufen, beim Beginn der Codierung erreicht wird. Dies wird durch einen Terminator TR erreicht, der aufgrund der gespeicherten Information eine Terminierungssequenz $\underline{Z} = Z_1, Z_2$ der Länge M = 2 Bits — entsprechend der Anzahl der Speicherstufen K1, K2 des Coders — erzeugt, die in Kombination mit den rückgeführten Bits alle Speicherstufen in den Zustand log. Null bringt.

Die Terminierungssequenz kann aus den gespeicherten Bits errechnet werden, die bei einer Modulo-2-Addition zu Null ergänzt werden. Die Terminierungssequenz kann aber auch beispielsweise aus einem ROM kom-

plett aufgerufen werden. Nach der Terminierungsfolge wird die nächste Informationssequenz codiert.

Die ursprüngliche Coderate (Transferrate)

$$R = \frac{L}{2L} = \frac{1}{2}$$

wird nunmehr zu:

$$R_T = \frac{L}{2(L+M)}$$

mit L = Anzahl der Informationsbits je Informationssequenz und M = Länge der Terminierungssequenz bzw. Anzahl der Speicherstufen.

Allgemein gilt:

$$R_T = \frac{L}{N(L+M)}$$

mit 1/N = Coderate ohne Termination.

Die durch die Terminierung etwas geringer gewordene Coderate läßt sich, falls nötig, durch als Punktierung bezeichnete Selektion von Codesymbolen aus der ausgegebenen Codefolge ausgleichen.

Der Decoder weist eine dem Coder verwandte Struktur auf. Der Decodierungsalgorithmus erfolgt entsprechend einem Trellisdiagramm, wie es beispielsweise in "Digital Communication", 2nd Edition von John G. Proakis, McGraw-Hill, Inc. auf Seite 447 dargestellt ist.

Da die Codierung nicht mit dem letzten Informationsbits abgebrochen, sondern um die Terminierungssequenz verlängert wird, wobei selbstverständlich auch der Decodierer den Sollzustand des Coders kennt, wird der Fehlerschutz für die letzten Informationsbits vergrößert.

In **Fig.** 2 ist ein Schema zur "Turbo-Codierung" angegeben, bei dem mehrere Komponentencoder COD1 bis CODn vorgesehen sind. Aus der in der Beschreibungseinleitung angegebenen Veröffentlichung und der Europäischen Patentanmeldung von C. Berrou ist dieses Verfahren beschrieben.

Über den Dateneingang **1** werden die Informationssequenz $\underline{I}$ über einen ersten (nicht unbedingt erforderlichen) Interleaver IV1 einem ersten Coder COD1 zugeführt. Der Interleaver hat die Aufgabe, die Informationsbits zu verwürfeln. Der Coder COD1 weist entsprechend dem Coder in **Fig.** 1 einen Umschalter SW1 und einen Terminator TR1 auf.

An den Ausgängen des ersten Coders COD1 werden Codefolgen $\underline{X}_1 = \underline{I}_1, \underline{Z}_1, \underline{P}_{1,1}, \ldots, \underline{P}_{1,K1}$ von Bits der Informationssequenz(en) $\underline{I}$ und der Terminierungssequenz(en) $\underline{Z}$ sowie Kontrollsequenz(en) $\underline{P}_{1,K1}$ ausgegeben. An den Informationsausgang **2** (oder die Informationsausgänge) des ersten Coders sind über weitere Interleaver IV2 bis IVn weitere Coder COD2 bis CODn angeschaltet, die wiederum Codefolgen $\underline{X}_2 = \underline{I}_2, \underline{Z}_2, \ldots, \underline{P}_{2,K2}$ bis $\underline{X}_n$ abgeben. Da durch den ersten Coder COD1 durch die Terminierungssequenz eine längere Codefolge $\underline{X}_1 = \underline{I}_1, \underline{Z}_1, \ldots \underline{P}_{1,K1}$ insbesondere eine längere "Informationssequenz" $\underline{I}_1, \underline{Z}_1$, entsteht, wird die Größe der an den Informationsausgang **2** angeschalteten Interleaver jetzt von der Anzahl L der ursprünglichen Informationssequenz und von der Länge M_{COD1} der Terminierungssequenz, insgesamt also durch $L + M_{COD1}$, bestimmt.

Die Bits der Terminierungssequenz $\underline{Z}_1$ können mit den Informationsbits verwürfelt werden oder nach dem jeweiligen Verwürfeln der Informationssequenz den weiteren Codern COD2, ..., CODn zugeführt werden.

Bei der Version nach **Fig.** 2 erfolgt die Terminierung des Trellis nur beim ersten Coder COD1. Sie kann aber auch bei den weiteren Komponentencodern COD2 bis CODn erfolgen, wodurch aber die Coderate weiter sinkt.

Durch Selektion von ausgegebenen Codesymbole in einer Auswahlschaltung SE wird die (bei der gewählten schematischen Darstellung) zunächst vervielfachte Datenrate reduziert, wobei in der Regel die Informationsbits $I_1, I_2, \ldots$ und die Zusatzbits der Terminierungsfolge nur einmal übertragen werden.

Häufig wird durch Multiplexen eine serielle Codefolge $\underline{X}_s$ erzeugt wird.

Empfangsseitig werden selbstverständlich sowohl bei der Verwendung eines Coders als auch von mehreren Komponentencodern nach der Decodierung die wahrscheinlicheren Codesymbole für die ursprünglichen Informationssequenz $\underline{I}$ ausgegeben.

Die in der Regel in mehreren Durchgängen erfolgende Decodierung ist in den Veröffentlichungen von Berrou beschrieben.

In **Fig.** 3 ist eine Variante zur Erzeugung eines "Turbo-Codes" mit kaskadenmäßig angeordneten Komponentencodern dargestellt. An den "Informationsausgang" des ersten Coders COD1 ist wieder über den zugeordneten Interleaver IV2 der zweite Komponentencoder COD2 angeschaltet. In derselben Weise sind auch die weiteren Coder COD3, ..., CODn wiederum mit dem Ausgang des vorhergehenden Coders verbunden.

Die weiteren Coder COD2 und COD3 können ebenfalls Terminatoren enthalten. Dabei ist aber zu beachten, daß die Länge der ausgegebenen Codefolgen $\underline{X}_2$, $\underline{X}_{30}$, $\underline{X}_{n0}$ dann mit jedem weiteren Coder zunimmt.

In **Fig.** 4 ist eine andere Variante zur Erzeugung eines "Turbo-Codes" dargestellt. Mehrere Codierer COD1 bis CODn sind über zugeordnete Interleaver IV1 bis IVn an den Dateneingang **1** angeschaltet, dem die zu codierende Informationssequenz $\underline{I}$ zugeführt wird. Durch die Interleaver werden unterschiedliche Informationsfolgen I_1 bis I_n erzeugt, so daß die von den jeweils einen eigenen Terminator TR1 bis TRn aufweisenden Codern ausgegebenen Codefolgen $\underline{X}_1, \underline{X}_2, \ldots \underline{X}_n$ ebenfalls unterschiedlich sind.

Auch hier kann die durch die Terminierungssequenzen verringerte Coderate, falls nötig, durch weitergehende Selektion Codesymbolen der ausgegebenen Codefolgen ausgeglichen werden.

Patentansprüche

1. Verfahren zur Terminierung des Trellis bei rekursiven systematischen Faltungscodes,
bei dem nach Codierung einer Informationssequenz (I) in einem Coder (COD) diesem eine vom Zustand seines Gedächtnisses (K1, K2) abhängige Terminierungssequenz (Z) erzeugt wird, die dem Codereingang zugeführt wird und dessen Gedächt-

nis (K1, K2) in einem bestimmten Sollzustand bringt, und bei dem
die Terminierungssequenz (Z) und die zusätzlich erzeugten Kontrollbits ebenfalls übertragen werden.

2. Verfahren nach Anspruch 1, dadurch gekennzeichnet, daß der Sollzustand des Coder-Gedächtnisses (K1, K2) derart eingestellt wird, daß sie mit dem Anfangszustand des Coder-Gedächtnisses zu Beginn der Codierung der Informationssequenz (I) übereinstimmt.

3. Verfahren nach Anspruch 1, dadurch gekennzeichnet, daß als Anfangszustand des Coder-Gedächtnisses (K1, K2) der Zustand "log. Null" oder "log. Eins" für alle Speicherstufen (K1, K2) eingestellt wird.

4. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß jeweils das nächste dem Coder zugeführte Bit der Terminierungssequenz (Z) aus den Zuständen des Coder-Gedächtnisses (K1, K2) errechnet wird.

5. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß Informationssequenzen (I) gleicher Länge (L) übertragen werden.

6. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß mehrere rekursive systematische Faltungscodes als Komponentencodes für eine Turbo-Codierung erzeugt werden.

7. Verfahren nach einem der-vorhergehenden Ansprüche, dadurch gekennzeichnet,
daß ein erster Komponentencode ($\underline{X}_1$) in einem ersten Coder (COD1) erzeugt wird,
daß eine Terminierung des Trellis durch eine Terminierungssequenz ($\underline{Z}_1$) in diesem Coder (COD1) erfolgt,
daß aus der Informationssequenz ($\underline{I}_1$) und der anschließenden Terminierungssequenz ($\underline{Z}_1$) nach dem Interleaven dieser Folgen unter Verwendung weiterer Komponentencoder (COD2) mindestens eine weiter Codefolge ($\underline{X}_1, \ldots \underline{X}_n$) generiert wird.

8. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Coderate der zu übertragenden Codesymbole durch Selektion von erzeugten Codesymbolen erhöht wird.

9. Coder zur Erzeugung eines rekursiven systematischen Codes, dadurch gekennzeichnet, daß ein Terminator (TR) vorgesehen ist, der nach Codierung einer Informationssequenz (I) eine vom Zustand des Coder-Gedächtnisses (K1, K2) abhängige Terminierungssequenz (Z) erzeugt, die dem Codereingang zugeführt wird und das Coder-Gedächtnis (K1, K2) in einen bestimmten Sollzustand (z. B. log.0, log.1) bringt.

10. Coder nach Anspruch 10, dadurch gekennzeichnet, daß er (COD1) und mindestens ein weiterer Coder (COD2, ... CODn) zur Erzeugung von rekursiven systematischen Komponentencodes eines Turbo-Codes vorgesehen sind.

11. Coder nach Anspruch 9 oder 10, dadurch gekennzeichnet, daß an seinem Informationsausgang (2) über Interleaver (IV2, ... IVn) die weiteren Coder (COD2, ... CODn) angeschaltet sind.

12. Coder nach Anspruch 9 oder 10, dadurch gekennzeichnet, daß die weiteren Coder (COD2, ... CODn) kaskadenmäßig jeweils über einen zugeordneten Interleaver (IV2, ... IVn) an den Informationsausgang (2, ...) des vorhergehenden Coders angeschaltet sind.

Hierzu 3 Seite(n) Zeichnungen

Nummer: DE 195 20 987 A1
Int. Cl.[6]: H 03 M 13/12
Offenlegungstag: 12. Dezember 1996

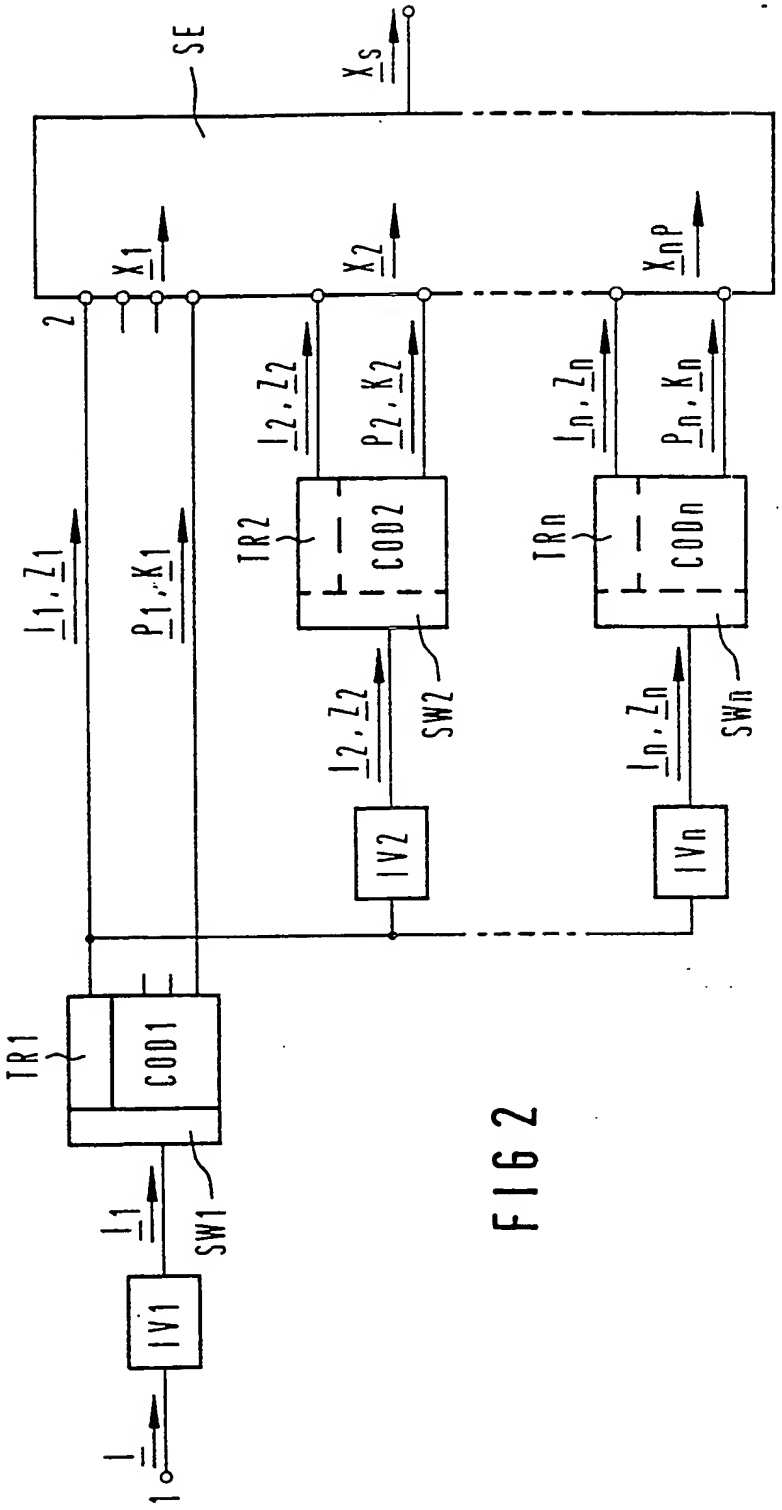


FIG 2

Nummer: DE 195 20 967 A1
Int. Cl.[6]: H 03 M 13/12
Offenlegungstag: 12. Dezember 1996

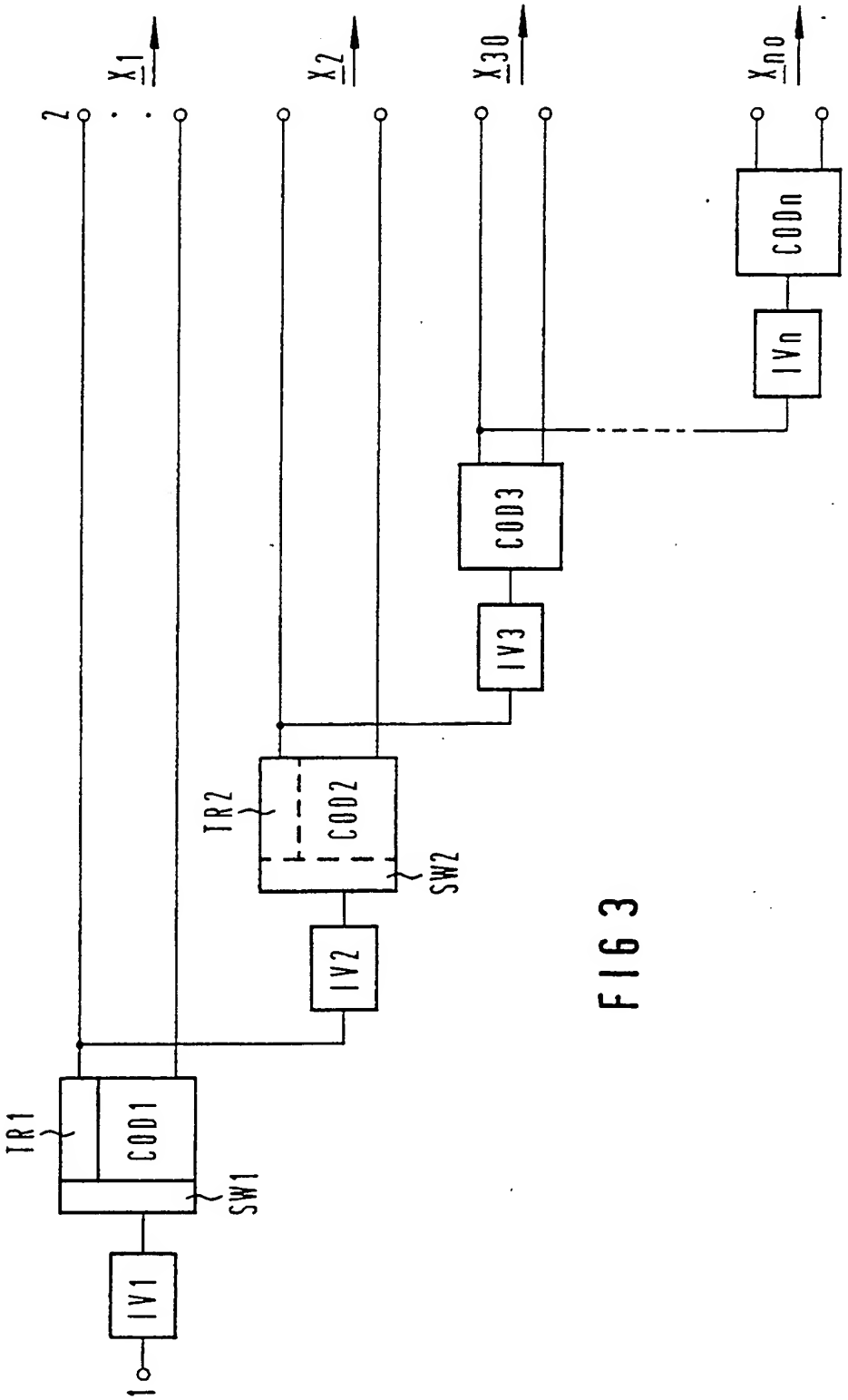


FIG 3

- Leerseite -

Nummer: DE 195 20 987 A1
Int. Cl.[6]: H 03 M 13/12
Offenlegungstag: 12. Dezember 1996

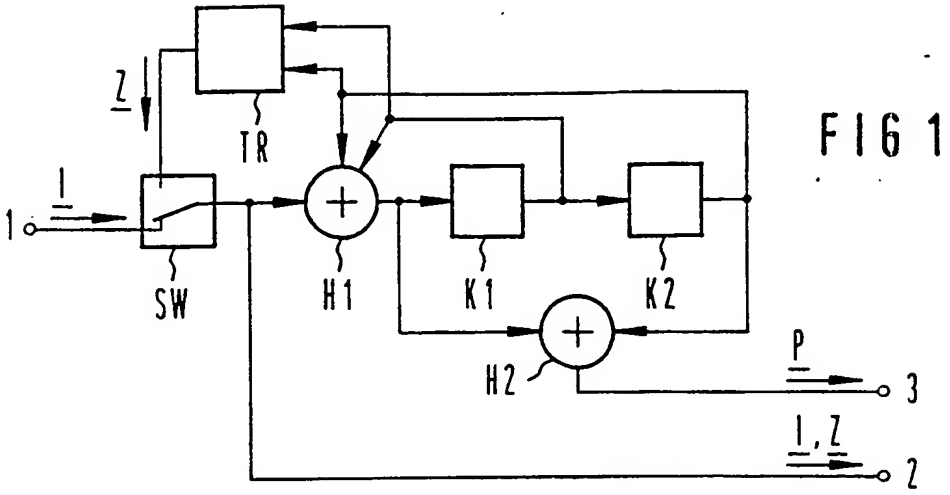
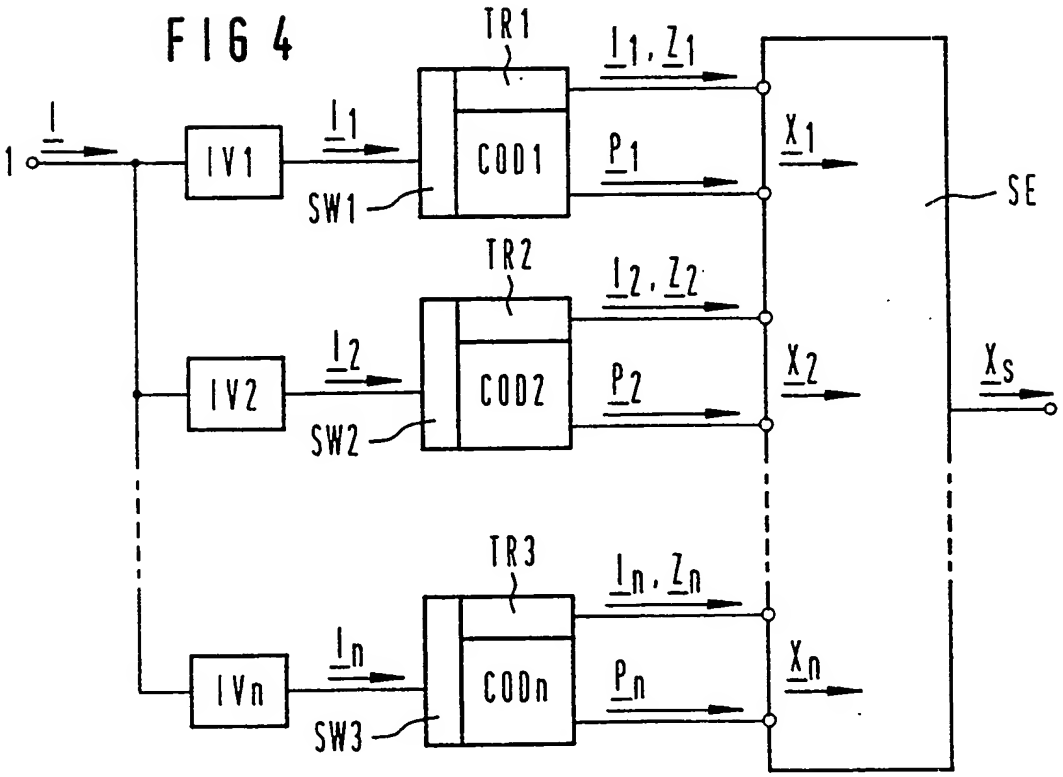


FIG 1

FIG 4



TRANSLATION: DE 195 20 987 A1

Applicant:	Siemens AG (DE)
Inventors:	Hagenauer, Joachim Burkert, Frank
Application date:	June 8, 1995
Publication date:	December 12, 1996
Title:	Method for terminating trellis in recursive systematic convolutional codes

Abstract

The coder (COD) contains a terminator (TR) for generating, after encoding an information sequence (I) based on the coder memory (K1, K2), a termination sequence (Z) which takes the coder memory (K1, K2) into a desired target state.

Description

The invention relates to a method for termination of the trellis in recursive systematic convolutional codes, and to a coder adapted to carry out the method.

Various encoding methods are being used for the purposes of data protection. In these methods, check bits are derived from the information bits to be transmitted, which check bits are equally transmitted. With the aid of these check bits it is possible to recognize and correct falsified information bits on the receiver side.

In convolutional codes, so-called tail biting methods were developed so as to enhance error protection for the last information bits to be encoded, which decreases in the absence of additional measures. This method is indicated in IEEE Trans. on Comm., Vol. COM-34, No.2, 1986 by H.H. Ma, J.K. Wolf, "On Tail Biting Convolutional Codes."

The known tail biting methods for non-recursive convolutional codes may, however, not be transferred to recursive systematic convolutional codes. Owing to its properties, however, this code group is of extraordinary importance for multi-component codes, the so-called "turbo-codes". This code group is described in detail in the following literature:

- C. Berrou, "Near Shannon limit error-correcting and decoding: Turbo-Codes (1)", Proc. ICC '93, May 1993;
- Demande de brevet européen, N° de publication: 0 511 141 A1, Inventeur: C.Berrou, "Procédé de codage correcteur d'erreurs à moins deux codages convolutifs systématiques en parallèle, procéde de décodage itératif, module de décodage et décodeur correspondants"
- J.Hagenauer et al, "Iterative ("Turbo") decoding of systematic convolutional codes with MAP and SOVA algorithms", ITG Fachtagung "Codierung", München, Okt. 1994
- J. Hagenauer, L. Papke, "Decoding "Turbo"-Codes with the Soft Output Viterbi Algorithm (SOVA)", 1994 International Symposium on information theory, Trondheim, 1994.

From IEEE, Globecom 1994, pages 1298 to 1303, Robertson:
"Illuminating the Structure of code and decoder of parallel concatenated recursive systematic (Turbo) codes", the "trellis termination" is also described for recursive codes, however without indication of a realization.

In the "zero tail method" a coder for a non-recursive convolutional code is supplied, after coding of the information bits, with a sequence of overhead bits forcing it into the desired target state, i.e., the trellis is terminated. This circumstance is jointly assessed by the decoder, whereby error protection for the final information bits of a data block is enhanced.

It is the object of the invention to indicate an easily realized method for terminating the trellis in recursive systematic convolutional codes. Moreover a suitable coder is to be indicated.

This object is achieved through the method indicated in claim 1.

In an independent claim a suitable coder is disclosed.

Advantageous developments of the method and of the coder are specified in the appended claims.

The method gains particular simplicity if all memory stages of the coder have the same information, log.null or log.one, in the target state. This condition may match the initial state at the beginning of encoding a data block.

Termination becomes particularly simple if the respective overhead bit of the termination sequence next to be supplied to the coder is calculated starting out from the states of the coder memory, e.g. of memory stages.

In realizing a turbo-code, termination of the trellis may be performed in one or several component coders, but also in all component coders.

The inventive method shall be described in more detail by referring to practical examples.

Fig. 1 shows a coder for a recursive systematic code with terminator,

Fig. 2 shows a circuit diagram for realization of multi-component codes (turbo-codes) with terminator,

Fig. 3 shows a first variant for realization of multi-component codes (turbo-codes) with terminator, and

Fig. 4 shows another variant for realization of multi-component codes (turbo-codes) with terminator.

In **Fig. 1** a coder for a recursive systematic code having two binary memory stages K1, K2 and two modulo-2 adders H1 and H2 is represented. Through a data input 1 and a changeover switch SW one respective information sequence $\underline{I} = I_1, I_2, \ldots I_L$ arrives bitwise at the information output 2, and simultaneously at an input of the first modulo-2 adder H1 which moreover is fed the bits present in memory stages K1, K2, the coder memory. The result of the modulo-2 addition is supplied to the data input of the first memory stage K1. Through another modulo-2 addition of the modulo-2 sum at the output of the first modulo-2 adder H1 and the information present at the output from the second memory stage K2, check bits P are generated and output at the control output 3. The code symbols (bits) present at outputs 2 and 3 are generally emitted bitwise in interleaved condition. The coder operates in a known manner by using a bit timing signal which is not represented in this circuit diagram.

Once the information of one data block has been coded, it the problem now is to modify the coder memory, i.e. M data items stored in memory stages K1 and K2 -- altogether 2^M different variations are possible in binary memories -- in such a way that a particular target state, e.g. the initial state "log.null", is attained for all memory stages at the beginning of coding. This is achieved with the aid of a terminator TR which, based on the stored information, generates a termination sequence $\underline{Z} = Z_1, Z_2$ of length M = 2 bits - corresponding to the number of memory stages K1, K2 of the coder - which, in combination with the fed-back bits, takes all memory stages into the state log.null.

The termination sequence may be calculated from the stored bits which are supplemented to become zero in a modulo-2 addition. The termination sequence may, however, for example also be called up complete from a ROM. After the termination sequence the next information sequence is coded.

The original code rate (transfer rate)

$$R = \frac{L}{2L} = \frac{1}{2}$$

now becomes:

$$R_T = \frac{L}{2(L+M)}$$

with L = number of information bits per information sequence and M = length of the termination sequence or number of memory stages.

In general:

$$R_T = \frac{L}{N(L+M)}$$

with 1/N = code rate without termination.

The code rate, reduced somewhat on account of the termination, may be compensated, where necessary, by selection of code symbols from the output code sequence that is referred to as puncturing.

The decoder has a structure similar to that of the coder. The decoding algorithm is performed in accordance with a trellis diagram, for example in accordance with the representation in "Digital Communication", 2nd Edition by John G. Proakis, McGraw-Hill, Inc. on page 447.

Error protection for the last information bits is enhanced inasmuchc as coding is not broken off concurrently with the last information bits but is prolonged by the termination sequence, and the decoder does, of course, also know the coder's target state.

In **Fig. 2** a schematic representation for "turbo-coding" is given where several component coders COD1 to CODn are provided. This method is described by the publication indicated in the introductory portion of the description and by the European patent application by C. Berrou.

Through data input 1 the information sequence $\underline{I}$ is supplied via a first (not definitely necessary) interleaver IV1 to a first coder COD1. The interleaver has the task of scrambling the information bits. The coder COD1 includes in correspondence with the coder in **Fig. 1** a changeover switch SW1 and a terminator TR1.

At the outputs of the first coder COD1, code sequences $\underline{X}_1 = \underline{I}_1, \underline{Z}_1, \underline{P}_{1,1}, \ldots, \underline{P}_{1,K1}$ of bits of the information sequence(s) $\underline{I}$ and of the termination sequence(s) $\underline{Z}$, as well as control sequence(s) $\underline{P}_{1,K1}$ are output. To the information output 2 (or to the information outputs) of the first coder, additional coders COD2 to CODn are connected via further interleavers IV2 to IVn, which additional coders in turn output code sequences $\underline{X}_2 = \underline{I}_2, \underline{Z}_2, \ldots, \underline{P}_{2,K2}$ to $\underline{X}_n$. As a longer code sequence $\underline{X}_1 = \underline{I}_1, \underline{Z}_1, \ldots \underline{P}_{1,K1}$, in particular a longer "information sequence" $\underline{I}_1, \underline{Z}_1$ is generated by the first coder COD1 due to the termination sequence, the size of the interleavers connected to the information output 2 is now determined by the number L of the original information sequence and by the length M_{COD1} of the termination sequence, thus in total by $L + M_{COD1}$.

The bits of the termination sequence $\underline{Z}_1$ may be scrambled with the information bits or be supplied to the further coders COD2, ... CODn after a respective scrambling of the information sequence.

In the version in accordance with Fig. 2, termination of the trellis only takes place at the first coder COD1. It may, however, also take place at the further component coders COD2 to CODn, whereby the code rate is further decreased, however.

Through selection of output code symbols in a selection circuit SE the data rate, initially multiplied (in the selected schematic representation), is reduced, with the information bits $\underline{I}_1$, $\underline{I}_2$, ... and the overhead bit of the termination sequence as a rule being transmitted only once.

Frequently a serial code sequence $\underline{X}_S$ is generated by multiplexing.

On the receiver side, the higher-probability code symbols for the original information sequence $\underline{I}$ are, of course, output after decoding both when one coder is used and when several component coders are used.

Decoding, as a general rule taking place in several runs, is described in the publications by Berrou.

In **Fig. 3** a variant for generation of a "turbo-code" with cascaded component coders is represented. The second component coder COD2 is again connected to the "information output" of the first coder COD1 via the associated interleaver IV2. In the same way, the additional coders COD3, ..., CODn are in turn connected with the output of the preceding coder.

The additional coders COD2 and COD3 may also contain terminators. It should here be noted, however, that the length of the output code sequences $\underline{X}_2$, $\underline{X}_{30}$, $\underline{X}_{n0}$ then increases with each additional coder.

In **Fig. 4** another variant for generation of a "turbo-code" is represented. Several coders COD1 to CODn are connected via associated interleavers IV1 to IVn to the data input 1 to which the information sequence $\underline{I}$ to be coded is supplied. By the interleavers different information sequences $\underline{I}_1$ to $\underline{I}_n$ are generated, so that the code sequences $\underline{X}_1$,

X_2, ... X_n output by coders each having their own terminator TR1 to TRn are also different.

Here, too, the code rate reduced by the termination sequences may, where necessary, be compensated by further-going selection of code symbols of the output code sequences.

Claims

1. A method for terminating trellis in recursive systematic convolutional codes,

wherein after coding an information sequence ($\underline{I}$) in a coder (COD) , a termination sequence ($\underline{Z}$) is generated by the latter in accordance with a state of the memory (K1, K2) thereof, which termination sequence is supplied to the coder input and takes the memory (K1, K2) thereof into a particular target state, and wherein

the termination sequence ($\underline{Z}$) and the additionally generated check bits are also transmitted.

2. The method in accordance with claim 1, characterized in that the target state of the coder memory (K1, K2) is set such that it matches the initial state of the coder memory at the beginning of coding the information sequence ($\underline{I}$).

3. The method in accordance with claim 1, characterized in that as the initial state of the coder memory (K1, K2), the state "log.null" or "log.one" is set for all memory stages (K1, K2).

4. The method in accordance with any one of the preceding claims, characterized in that the respective bit of the termination sequence ($\underline{Z}$) next to be supplied to the coder is calculated starting out from the states of the coder memory (K1, K2).

5. The method in accordance with any one of the preceding claims, characterized in that information sequences ($\underline{I}$) having identical lengths (L) are transmitted.

6. The method in accordance with any one of the preceding claims, characterized in that several recursive systematic convolutional codes are generated as component codes for turbo coding.

7. The method in accordance with any one of the preceding claims, characterized in that a first component code ($\underline{X}_1$) is generated in a first coder (COD1),

termination of the trellis takes place in this coder (COD1) with the aid of a termination sequence ($\underline{Z}1$),

at least one further code sequence ($\underline{X}_1$, ... $\underline{X}_n$) is generated from the information sequence ($\underline{I}_1$) and from the subsequent termination sequence ($\underline{Z}_1$) after interleaving of these sequences by using further component coders (COD2).

8. The method in accordance with any one of the preceding claims, characterized in that the code rate of the code symbols to be transmitted is increased through selection of generated code symbols.

9. A coder for generating a recursive systematic code, characterized in that a terminator (TR) is provided which, after coding of an information sequence (I), generates a termination sequence (Z) which depends on a state of the coder memory (K1, K2) and which is supplied to the coder input and takes the coder memory (K1, K2) to a particular target state (such as log.0, log.1).

10. The coder in accordance with claim 10, characterized in that the said coder (COD1) and at least one further coder (COD2, ... CODn) for generating recursive systematic component codes of a turbo-code are provided.

11. The coder in accordance with claim 9 or 10, characterized in that the further coders (COD2, ... CODn) are connected to the information output (2) thereof via interleavers (IV2, ... IVn) .

12. The coder in accordance with claim 9 or 10, characterized in that the additional coders (COD2, ... CODn) are each connected in cascade connection to the information output (2, ...) of the preceding coder via respective associated interleavers (IV2, ... IVn).